APPARATUS AND METHOD FOR RECORDING FIRMWARE IN COMPUTER SYSTEM

BACKGROUND OF THE INVENTION

1. Field of the Invention

[1] The present invention relates to an apparatus and method for recording firmware in a computer system.

2. Background of the Related Art

[2] In general, a computer system includes an embedded controller, namely a micro-controller for controlling inputs of a keyboard and a mouse, supplying or intercepting power, and turning on/off the system. When power is applied to the computer system, the micro-controller examines the state of the respective components composing the computer system on the basis of firmware recorded on a ROM, such as a flash memory, and performs a POST (Power On Self Test) process for initialization in order for the computer system to operate. After the micro-controller normal operation of the POST process, an operation system starts to control the system.

[3] During the POST process, a system BIOS (Basic Input Output System) is operated through the communication with the micro-controller. The micro-controller controls LCD, brightness control or volume, battery and thermal information. The system BIOS is stored in the system ROM such as a flash ROM, and a keyboard BIOS is stored in an H8 keyboard controller, which is a micro-controller.

[4] The process of recording firmware is generally performed in a system development step. As shown in Fig. 1, after the process starts, a control program for controlling the recording process is loaded to a RAM (step S12) according to a firmware recording command (step S11). If the firmware recording command is not received, the check for the firmware recording command is repeated (step S11). The control program controls deleting the whole contents of the ROM (step S13), and consecutively recording new firmware having initialization information and other functions on the ROM (step S14). Then. The process is completed.

[5] However, if the firmware damaged because of a code bug or update error is recorded on the ROM, the system may stop on a log screen during the POST process of the system BIOS, or may not be responsive to system power-on. To solve the foregoing problems, the micro-controller having the ROM in which the damaged firmware has been recorded on must physically replace the damaged firmware with a normal one. In this case, a main board, connections, devices coupled thereto or an operator may be damaged because of a high difficulty of the corresponding replacement and/or soldering process. Such replacements can also cause waste of money, time and manpower, which results in low productivity and increased costs.

[6] The above references are incorporated by reference herein where appropriate for appropriate teachings of additional or alternative details, features and/or technical background.

SUMMARY OF THE INVENTION

[7] An object of the invention is to solve at least the above problems and/or disadvantages and to provide at least the advantages described hereinafter.

[8] Another object of the present invention is to provide an apparatus and method for recording firmware in a computer system that can divide a storage device (e.g., ROM) on which firmware is recorded into a main block and an auxiliary block. Firmware having entire system functions and firmware for updating the firmware of the main block can be recorded in the main and auxiliary blocks, respectively. The firmware in the auxiliary block for updating the firmware of the main block can be system recovery function routine firmware such as Power on, POST and Flash Routines. Correcting or updating the damaged firmware of the main block can be done using the recovery function routine firmware of the auxiliary block without performing a physical operation for replacing a micro-controller or the like.

[9] Another object of the present invention is to provide an apparatus and method for recording firmware in a computer system that can re-record firmware without physical replacement.

[10] Another object of the present invention is to provide an apparatus and method for recording firmware in a computer system that can perform a POST process even when firmware of a main block is damaged, by recording firmware for performing the POST process on a POST proof block, which can be an auxiliary block.

[11] The foregoing and other objects and advantages can be realized in a whole or in part by providing an apparatus for recording firmware in a computer system that includes a first memory divided into at least two blocks, wherein each of the blocks store system control programs and a second memory that temporarily stores the programs stored in the first memory used to selectively update the first memory.

[12] To further achieve at least the above objects in whole or in part a method for recording firmware in a computer system is provided that includes dividing an area for storing system control programs into at least two blocks, recording a first program on a first block, and recording a second program for selectively updating the first program on a second block.

[13] Additional advantages, objects, and features of the invention will be set forth in part in the description which follows and in part will become apparent to those having ordinary skill in the art upon examination of the following or may be learned from practice of the invention. The objects and advantages of the invention may be realized and attained as particularly pointed out in the appended claims

BRIEF DESCRIPTION OF THE DRAWINGS

[14] The invention will be described in detail with reference to the following drawings in which like reference numerals refer to like elements wherein:

[15] Fig. 1 is a flowchart showing a related art method for recording firmware in a computer system;

[16] Figs. 2(A)–2(D) are a block diagrams illustrating a micro-controller in accordance with a preferred embodiment of the present invention;

[17] Fig. 3 is a flowchart showing a preferred embodiment of a method for recording the firmware in a computer system in accordance with the present invention;

[18] Fig. 4 is a flowchart showing exemplary operations for shifting to a block that will be controlled in the method for recording the firmware in the computer system; and

[19] Fig. 5 is a flowchart showing another preferred embodiment of a method for recording the firmware in the computer system in accordance with the present invention.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

[20] Preferred embodiments of an apparatus and methods for system recovery or recording firmware in a computer system according to the present invention will now be described. According to preferred embodiments of the present invention, an apparatus and method for system recovery in a computer can include a ROM or the like in which firmware is recorded preferably logically divided into two blocks. In general, a process of dividing the ROM into two blocks can be physically performed, but can also be performed using various methods, for example logically.

[21] One block of the divided firmware can be a main block in which main firmware for performing the whole functions is recorded, and the other block can be an auxiliary block in which system recovery function routine firmware, for example, firmware for performing the POST function and for performing the main firmware, such as Power On,

POST and Flash Routines as DOS Routine) is recorded. Thus, the auxiliary block can be a POST-proof block. As used herein, the terminology "firmware" stored using a ROM, can include a type of program such as an application program and thus can be used with the same meaning as the program in the related art and preferred embodiments of the present invention.

[22] When an error occurs in the firmware recorded on the main block, or when the firmware is damaged, a micro-controller of the computer system cannot be normally operated. According to preferred embodiments, normal or updated firmware can be recorded on a corresponding block (e.g., damaged firmware) of the ROM according to a firmware update, re-recording command or the like.

[23] Fig. 2(A) is a block diagram illustrating a preferred embodiment of a micro-controller in accordance with the present invention. Preferred embodiments of methods for recording firmware in the computer system can be applied to the micro-controller shown in Figure 2(A) in accordance with the present invention as described hereafter. As shown in Fig. 2(A), a micro-controller 10 is operably coupled to an application program and can include RAM 20, and ROM 30 having main block 301 and an auxiliary block 302. Control software for recording new firmware under the control of an application program, or new firmware that will be recorded on the main block 301 or a POST proof block 302 under the control of the control software is preferably loaded to a RAM 20. The new firmware can be recorded on a ROM 30, which can be logically divided into the main block 301 and the POST proof block 302.

[24] When the main block 301 is damaged, the system is powered on by a block (e.g., power on routines) stored in the auxiliary block, (e.g. POST proof block 302), and a system BIOS performs a POST process, and to preferably enter into a DOS mode. A user can perform a flash process in the DOS mode according to an application program to update the damaged block of the main block 301.

[25] As shown in Figs. 2(A)-2(D), the application program can denote a flash utility program, the control software can denote an update routine, and the new firmware can denote a recovery file for receiving the damaged block. The application program, control software and new firmware are preferably stored in one storing device (for example, floppy disk) in the form of files in the DOS mode, so that the user can recover the system by using the disk or remote communication.

[26] An exemplary process of updating an initialization firmware such as damaged micro-controller 10 by using a previously individually stored recovery function block will now be described using Figs. 2(A)-2(D). Fig. 2(A) shows the structure of the ROM 30 logically divided into two blocks for respectively storing firmware, and the RAM 20 can record an execute program of the micro-controller 10 or various processing data generated by the execute program. In this example in Fig. 2(A), the damaged firmware has not been updated.

[27] The main block 301 stores data for performing general functions of the micro-controller 10, and the POST proof block 302 stores data necessary for the POST process. Accordingly, when the main block 301 is damaged, it can be recovered by using

the POST proof block 302, and when the POST proof block 302 is damaged, it can be recovered by using the main block 301.

[28] For example, when the main block 301 is damaged, the system is powered on by using information (e.g., power-on routines) of the individual block, namely the POST proof block 302, and then the system BIOS can perform an initial booting process up to the DOS mode by using the information of the POST proof block 302. Processes of updating the damaged block will be described below.

[29] Fig. 2(B) shows a process of loading the update routine (e.g., control software) for recording new firmware to the RAM 20 preferably by the flash utility program, when the user inputs an update command. The flash utility program preferably reads the update routine, and selects a loading path of the update routine according to the state of the firmware recorded on the main block 301 or the POST proof block 302. If the main block 301 has control, the update routine can be loaded through the firmware recorded on the main block 301, and if the POST proof block 302 has control, the update routine can be loaded through the firmware recorded on the POST proof block 302.

[30] When the micro-controller 10 is reset and driven, control is preferably selected by, for example, combinations of signals inputted to the micro-controller 10 in an initial reset routine. That is, when the previously-set signal combinations are inputted, the firmware of the POST proof block 302 has control. Otherwise, the firmware of the main block 301 has controllability. Thus, the non-failed block of the main block 301 and the POST proof

block 302 is set up to have control for loading the update routine to the RAM 20 for updating.

[31] Fig. 2(C) shows a process of deleting the whole contents of a failed object block to which new firmware will be recorded for recovery, when the object block is designated according to the flash utility program after the update routine (e.g., control software) is loaded to the RAM 20. When the main block 301 is designated as the object block preferably by the user according to the flash utility program, the update routine optionally or preferably deletes the contents of the main block 301. When the POST proof block 302 is designated as the object block preferably according to the flash utility program, the update routine deletes the contents of the POST proof block 302.

[32] Fig. 2(D) shows a process of recording new firmware provided by the flash utility program on the object block after the contents of the object block have been deleted (e.g., all contents) by the update routine. The update routine then can record the new firmware on the object block. Thus, updating the failed object block by the new firmware can complete the recovery process.

[33] When the failed object block is the main block 301, preferably the whole contents of the main block 301 are deleted and updated by new firmware corresponding to the main block 301. When the POST proof block 302 has failed, the whole contents preferably of the POST proof block 302 are deleted and updated by new firmware corresponding to the POST proof block 302 using the main block 301.

[34] Fig. 3 is a flowchart showing a preferred embodiment of a method for recording the firmware in the computer system. The method of Fig 3 can be applied to and will be described with reference to the microcomputer of Fig. 2.

[35] As shown in Fig. 3, after a process starts the block in which firmware (e.g., a system operating program) has not failed can obtain control and starts the update process (step S31). The control block in which new firmware will be updated can be designated by the user through use of the flash utility program (step S32).

[36] The update routine is loaded to the controller RAM preferably through the firmware of the block that will not be controlled (step S33). Then, the block that will be controlled is preferably updated (step S34), and the process ends.

[37] For example, if the main block is normally operated in the step for confirming the control block, the auxiliary block can be controlled, blanked and updated by the main block. On the other hand, when the main block is not normally operated, the main block can be controlled, blanked and updated by using the information of the auxiliary block.

[38] Fig. 4 is a flowchart showing an exemplary process for shifting control to the block that will update firmware when recording firmware in a computer system. As shown in Fig. 4, after a process starts it is determined whether the main block is being normally operated (e.g., confirming the control block, step S32 of Fig. 3) (step S41). When the main block is normally operated, the firmware recorded on the main block preferably containing control functions can be loaded to the RAM, for updating the corresponding block (e.g., auxiliary block) (step S42).

[39] However, if the main block is not normally operated, control is shifted to or obtained by the auxiliary block, and the following process can be performed to shift the control (e.g., control block). All power sources including a battery are preferably removed to reset a microcontroller (step S43). At least one previously-set key (e.g., function key+F4) is pressed, and simultaneously an AC power source is preferably supplied (step S44).

[40] A keyboard controller preferably detects the key pressed state and shifts control to the auxiliary block (step S45). From steps S42 and S45, the process can be completed.

[41] Fig. 5 is a flowchart showing another preferred embodiment of a method for recording firmware in a computer system in accordance with the present invention. The method of Fig 5 can be applied to and will be described with reference to the microcomputer of Fig. 2. However, the present invention is not intended to be so limited. After a process starts, when the user inputs the command for re-recording or updating the firmware of the ROM 30 of the micro-controller 10 of Fig. 2 (step S51), preferably the flash utility program, confirms the control block having controllability for updating the current firmware (step S52).

[42] Whether the control block exists in the main block 301 or the POST proof block 302 can then confirmed (step S53). The control block having control can be selected by combinations of hardware input signals in the initial reset routine or the like. If the previously-set signal combinations are inputted, the POST proof block 302 is preferably

selected as the control block, and if such signal combinations are not inputted, the main block 301 is selected as the control block.

[43] When one of the main block 301 and the POST proof block 302 is damaged, the other block can have control to update the damaged block. For example, when the main block fails on use, all power sources such as a battery are removed, and prescribed hardware can be determined. For example, when a key combination (e.g., function key+F4) is entered, AC power is applied, and a keyboard (e.g., H8 K/B) controller can detect the key pressed state and shift control to the auxiliary block. Accordingly, the functions set up in the auxiliary block, such as Power On, POST and Flash routines are performed, so that the system preferably enters into the DOS mode. As a result, the user can recover the system in the DOS mode or the like.

[44] The control block having control is preferably selected to update the main block when the auxiliary block has control. It prevents the main block and the auxiliary block from failing at the same time, and at least can maintain the auxiliary block in the normal state. Therefore, when the main block 301 has control, the contents of the main block 301 and the POST proof block 302 preferably can be re-recorded, and when the POST proof block 302 has control, the contents of the main block 301 can be re-recorded.

[45] When it is determined in step S53 that the control block exists in the POST proof block 302, the update routine for updating the firmware is loaded to the RAM 20 through the POST proof block 302 (step S54). Thereafter, when the main block 301 is designated as the block in which the firmware will be updated in by the flash utility program

(step S55), the update routine loaded to the RAM 20 deletes contents (e.g., all contents) recorded on the main block 301 (step S56) and records the new firmware on the corresponding block (e.g., main block 301) of the ROM 30 according to the flash utility program (step S57). Accordingly, when the main block 301 has failed, it can be updated by software, instead of replacing the whole micro-controller 10 or ROM containing the damaged firmware.

[46] On the other hand, when the control block is determined to be the main block 301 (step S53), the update routine is loaded to the RAM 20 through the main block 301 (step S58). Thereafter, the loaded update routine preferably confirms a target block that will be updated (step S59), and decides whether the main block 301 or the POST proof block 302 is the target block (step S60).

[47] When the main block 301 is determined to be the target block (step S60), the update routine loaded to the RAM 20 and the process can jump to step S55 to perform the update recording process. If the POST proof block 302 is the target block, the update routine preferably designates the POST proof block 302 as the block in which the firmware will be updated by the flash utility program (step S61). Thereafter, the update routine can delete the whole contents of the POST proof block 302 (step S62), and record new firmware on the POST proof block 302 (step S63).

[48] As described above, the information of the main block 301 and the POST proof block 302 can be updated by new firmware. Further, even if one of the two blocks is

damaged, it can be recovered by new firmware. It is thus possible to update the damaged block with software, instead of replacing the micro-controller.

[49] In addition, a code block for successively performing the POST process can be individually formed in the firmware of the micro-controller like a keyboard/embedded controller of the PC (portable computer) system, so that the firmware can be updated in any case without performing the physical operation for replacing the micro-controller. Preferably, when the firmware of the main block 301 is damaged, its embedded controller cannot detect when the user presses a power button. However, the system can be powered on by using the Power-On routine of the POST proof block 302, and thus the damaged firmware can be updated.

[50] A prescribed or minimum-sized firmware block for successively performing the POST process can be formed and prevented from being deleted and recorded in the update process. Accordingly, the firmware for successively performing the POST process can always exist in the micro-controller (e.g., for future use).

[51] As described above, preferred embodiments of a method and apparatus for firmware recovery have various advantages. Preferred embodiments can update damaged firmware during the production (e.g., manufacturing) without physical operations, reduces time consumption for handling errors during the development, and can directly cope with update errors in a user environment. Further, preferred embodiments for recording the firmware in the computer system can logically divide the ROM in which firmware is recorded into a main block and an auxiliary block. Preferably, the firmware having the whole

functions are recorded on the main block and the firmware for updating the firmware of the main block are recorded on the auxiliary block to correct or update the firmware without performing the physical operation for replacing the micro-controller. Moreover, firmware in the computer system can perform the POST process even when the firmware of the main block is damaged, by recording the firmware for performing the POST process on the auxiliary block. In addition, errors generated during the production can be handled without replacing an IC, which remarkably cuts down labor and production cost and reduces time consumption for performing a soldering process during the development.

[52] The foregoing embodiments and advantages are merely exemplary and are not to be construed as limiting the present invention. The present teaching can be readily applied to other types of apparatuses. The description of the present invention is intended to be illustrative, and not to limit the scope of the claims. Many alternatives, modifications, and variations will be apparent to those skilled in the art. In the claims, means-plus-function clauses are intended to cover the structures described herein as performing the recited function and not only structural equivalents but also equivalent structures.